

Randomized Rounding for Routing and Covering Problems: Experiments and Improvements*

Benjamin Doerr¹, Marvin Künnemann², and Magnus Wahlström¹

¹ Max-Planck-Institut für Informatik, Saarbrücken, Germany

² Universität des Saarlandes, Saarbrücken, Germany

Abstract We investigate how the recently developed different approaches to generate randomized roundings satisfying disjoint cardinality constraints behave when used in two classical algorithmic problems, namely low-congestion routing in networks and max-coverage problems in hypergraphs. Based on our experiments, we also propose and investigate the following new ideas. For the low-congestion routing problems, we suggest to solve a second LP, which yields the same congestion, but aims at producing a solution that is easier to round. For the max-coverage instances, observing that the greedy heuristic also performs very good, we develop hybrid approaches, in the form of a strengthened method of derandomized rounding, and a simple greedy/rounding hybrid using greedy and LP-based rounding elements. Experiments show that these ideas significantly reduce the rounding errors.

For an important special case of max-coverage, namely unit disk max-domination, we also develop a PTAS. However, experiments show it less competitive than other approaches, except possibly for extremely high solution qualities.

1 Introduction

Randomized rounding is one of the core primitives in randomized algorithmics. In contrast to many deep theoretical results, only very little experimental knowledge exists, and almost no fine-tuning and other implementation advice exists. Such results became even more interesting, since in the last ten years two substantially different methods [20,6,7,4] extending the classical approach of Raghavan and Thompson [19,18] were developed.

The only experimental work on either classical randomized rounding or the new approaches seems to be [5]. It compares the different methods on randomly generated rounding problems. The purpose of this work is to extend these results to two less artificial problem classes, namely routing and covering problems. These problems are among the first ones for which randomized rounding has been proven (by theoretical means) to lead to good algorithms.

Randomized Rounding: Given an arbitrary real number x , we say that (the random variable) y is a randomized rounding of x , if y equals $\lfloor x \rfloor + 1$ with probability $\{x\} := x - \lfloor x \rfloor$ and $\lfloor x \rfloor$ otherwise. In simple words, the closer x is to the next larger integer, the higher the chance of being rounded up.

* Supported by the German Science Foundation (DFG) via its priority program (SPP) 1307 “Algorithm Engineering”, grant DO 749/4-2.

Randomized rounding builds on the simple observation that this keeps the expectation unchanged, that is, $E(y) = x$. This naturally extends to linear expressions. If y_1, \dots, y_n are randomized roundings of x_1, \dots, x_n and $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a linear function, then $E(f(y_1, \dots, y_n)) = f(x_1, \dots, x_n)$. If, in addition, the y_i are independent, then Chernoff bounds allow strong quantitative statements showing that with high probability, $f(y_1, \dots, y_n)$ is not far from its expectation. These two key facts allow to use randomized rounding in connection with integer linear programming. Two examples of this are given in the following sections.

The new aspect of the works [20,6,7,4] is that they allow to generate randomized roundings that satisfy certain cardinality constraints with probability one. That is, for certain sets I , we can prescribe that $\sum_{i \in I} y_i = \sum_{i \in I} x_i$, provided the right-hand side is integral. This can be done without giving in with the other properties—both methods generate randomized roundings that admit the same Chernoff bounds as independent randomized rounding.

For reasons of space, we cannot describe these rounding algorithms here. However, since in this paper we are mainly comparing them experimentally, the reader may treat them as black-box, keeping in mind only that they generate randomized roundings that look independent, but satisfy cardinality constraints.

We shall concentrate on disjoint cardinality constraints. This is the most common form of cardinality constraints. Also, the comparison of the methods is more interesting here, since all have the same time complexity.

Our Results: The aim of this work is to find out how well the different rounding approaches are suited to solve classical problems that are often attacked with LP-based methods, but also to try to find fine-tunings and alternative approaches.

As underlying problems we chose the classical low-congestion routing problem and the max-coverage problem. They are different in flavor since in the first, randomized rounding is used with a focus of exploiting Chernoff bounds in linear constraints and objective function. In the second, since the right-hand side of the inequalities is one, we cannot do so, but resort to accepting that a certain fraction of the vertices covered in the relaxation are not covered after rounding.

All our results indicate that generally the derandomized algorithms yield superior results. The increase run-time over the randomized versions usually is still negligible compared to the complexity of solving the LPs involved.

For the low-congestion routing problem, we regard routing requests placed randomly on a two-dimensional grid. We regard instances small enough to compute the optimum solutions via solving an integer linear program. We observe that randomized rounding with cardinality constraints obtains reasonably good solutions. Surprisingly, unlike in previous experiments [5], we observe that the bit-wise randomized approach of [4] produces better results than the tree-based one of [20].

The gap to the optimum is roughly halved if we use a derandomization of randomized rounding. Here, the derandomization of [20] obtained in [5] proved to be superior.

In an attempt to fine-tune randomized rounding, we propose solving a second LP which gives the same congestion in the relaxation, but aims at making the solution easier to round. While this naturally does not give improved theoretical guarantees, it yields a good reduction of the rounding errors, in particular, in combination with the

derandomization. This seems to be a fruitful approach whenever the additional cost of solving a second LP is admissible.

Our analysis of max-coverage shows that both randomized rounding and the greedy algorithm produce good results in general. However, for both there are instances showing the other behave much better. Analyzing the data produced in our experiments, we consider two paths to a hybrid approach. One way is to strengthen the derandomization to include a greedy component, as a gradient-based rounding; the other, complementary, is to spend part of the budget greedily, and solve the remaining instance via an LP- and randomized rounding-approach. Both hybrids perform better than either of the plain approaches; the gradient-based rounding performs particularly well. We also give a new way of rounding under weighted knapsack constraints, which is both significantly more practical and theoretically cleaner than the previously known method.

For a natural planar Euclidian version of the problem, we also give a PTAS. However, unlike for all other approaches used in this paper, the experimental results are not much better than the theoretical guarantees. Therefore, this is an alternative useful only if very good approximations are needed and if computation power is available plentiful.

2 Randomized Rounding for Low-Congestion Routing

The Low-Congestion Routing Problem in Networks. The low-congestion routing problem is one of the classical applications of randomized rounding [19]. In its simplest version, the objective is to route a number of requests through a given network, minimizing the maximum usage of an edge (“congestion”). Problems of this type found all kinds of applications, an early one being routing wires in gate arrays [9].

We regard the following basic variant, previously investigated in [19,18,11,20,7]. Given is a (directed) network $G = (V, E)$, together with k routing requests. Each consists of a source vertex s_i , a target vertex t_i and a demand r_i . The objective is to find, for each $i \in [k] := \{1, \dots, k\}$, a flow from s_i to t_i having flow value $r_i \in \mathbb{N}$, such that the congestion, that is, the maximum total flow over an edge, is minimized. This problem is easily formulated as integer linear program (ILP): We minimize the congestion C subject to the constraints

$$\forall e \in E : \sum_{i=1}^k x_{ie} \leq C \quad (1)$$

$$\forall i \in [k] : \sum_{e=(s_i,v) \in E} x_{ie} - \sum_{e=(v,s_i) \in E} x_{ie} = r_i \quad (2)$$

$$\forall i \in [k] \forall v \in V \setminus \{s_i, t_i\} : \sum_{e=(w,v) \in E} x_{ie} = \sum_{e=(v,w) \in E} x_{ie} \quad (3)$$

$$\forall i \in [k] \forall e \in E : x_{ie} \in \{0, 1\}. \quad (4)$$

We should add that [19] only regard the special case of all demands r_i being one, since randomized rounding respecting cardinality constraints with right-hand side greater than one was not available at that time. For an application with particular need for larger r_i , see the failure restoration problem in optical networks described in [7]. Also,

we should add that other authors in addition have edge capacities c_e and then minimize the relative congestion, but it is easily seen that this just replaces the C in the first type of constraints by $c_e C$.

Since already the case of unit demands is NP-complete, optimal solutions seem difficult to obtain. The common solution concept is to (i) solve the linear relaxation of the ILP and obtain a fractionally optimal solution (x^*, C^*) ; (ii) use *path stripping* to decompose each flow f_i encoded in x^* into a weighted sum $f_i = \sum_{P \in \mathcal{P}_i} y_{iP}^* f_P$, where \mathcal{P}_i is a finite set of s_i - t_i paths, for each $P \in \mathcal{P}_i$, f_P is the flow that has exactly one unit on each edge of P , and $y_{iP}^* \in [0, 1]$ —note that all this implies $\sum_{P \in \mathcal{P}_i} y_{iP}^* = r_i$; (iii) use randomized rounding to round all y_{iP}^* to $y_{iP} \in \{0, 1\}$ in such a way that the cardinality constraints $\sum_{P \in \mathcal{P}_i} y_{iP} = r_i$ are maintained. Now $\sum_{P \in \mathcal{P}_i} y_{iP} f_P$ is a flow from s_i to t_i with flow value r_i . These flows form a solution having a congestion of $C = \max_{e \in E} \sum_{i=1}^k \sum_{P \in \mathcal{P}_i, e \in P} y_{iP}$. Large deviation bounds show that this congestion is not far from the value C^* given by the relaxation [19,11]:

$$C = O\left(\frac{\log m}{\log(2 \log m / C^*)}\right), \text{ if } C^* \leq \log m;$$

$$C = C^* + O(\sqrt{C^* \log m}), \text{ if } C^* > \log m.$$

Recall that C^* is a lower bound for the optimal solution. Hence if C^* is not too small compared to m , then this approach gives very good approximation factors.

Algorithms Used. To approximately or exactly solve our test instances, we used the following algorithms. Whenever run-times permitted, we used the exact ILP-Solver ILOG CPLEX 11.0 to directly solve the ILP given by (1) to (4). All other approaches involve solving the linear relaxation of the ILP (for which again we used CPLEX) and then different rounding methods.

Since the ILP contains hard cardinality constraints, we cannot use the classical independent approach of Raghavan and Thompson (we did so, though, ignoring the cardinality constraints, to see if the cardinality constraints make rounding more difficult). There are two approaches to generate randomized roundings respecting cardinality constraints due to Srinivasan [20] and the first author [4]. Both can be derandomized [5,4], so that in total we have four rounding methods available. See the original papers or [5] for a more detailed discussion of these methods. All algorithms different from CPLEX were implemented in C/C++.

Experimental Set-up. To analyze the questions discussed in the introduction, we regarded the following type of instances. Motivated by the fact that many routing problems have a two-dimensional flavor (e.g., the wire routing problem of [9]), we chose a finite two-dimensional bi-directed grid. Note that this simple graph is far from trivial for routing, see, e.g., the thrilling one-turn routing conjecture in [9], which is, to the best of our knowledge still open.

We choose routing requests randomly as follows. Both s_i and t_i are chosen uniformly at random from V . To reduce otherwise the influence of randomness, we choose all demands as $r_i = 3$. We also tried placing the s_i, t_i uniformly at random on the outer border of the grid, but saw no significant differences.

The size n of the grid and the number of demands k was varied to create different instance sizes and densities. All numerical values reported are the averages over 100 runs. The times were measured on AMD dual processor 2.4 GHZ Opteron machines.

Analysis. A small subset of our results is presented in Table 1. For the grid sizes 5×5 , 10×10 and 15×15 together with 10 and 75 demands, we state the average values of the congestion of an optimal solution (line 1 of the table) and the amounts by which a solution computed by one of the four randomized rounding approaches (lines 2, 3, 5, and 6) is worse (in percent). Line 4 and 7 of the table refer to an improvement discussed in the subsequent subsection.

Particularly for instances that do show some congestion, we see that randomized rounding yields quite good solutions, much better than what the theoretical bounds would predict. Derandomization is clearly worth the small extra effort (cf. the run-times in Table 2), reducing the gap to the optimum by roughly a half.

Comparing the different methods, surprisingly, our experiments generally show that the bit-wise randomized rounding approach of [4] (line 3) produced slightly better rounding errors than the tree-based one of [20] (line 2). We do not understand this phenomenon currently. Among the derandomizations, as expected and similarly as for random instances [5], the derandomization of the tree-based approach of [20] given in [5] is superior to the derandomization of the bit-wise one in [4]. This is due to the iterative nature of the latter, see again [5].

We also used classical independent randomized rounding. Clearly, this does not produce feasible solutions in most cases. However, even ignoring this issue, we also observed that we typically have slightly larger congestions (e.g. in the sparse instance of 10 demands in a 15×15 grid, independent rounding lead to a congestion of 3.19 compared to congestions of 2.80 and 2.85 for the randomized approaches of [4] and [20]).

Run-times consumed by the different stages are mainly given in Table 2. All randomized rounding stages for each instance took less than 0.02 seconds. Less than a tenth of this is the time needed for the path-stripping in each instance. Hence these numbers are not given in the table. From the table, we see that the bit-wise derandomization takes about 20 times longer than the tree-based one, but both numbers are greatly dominated by the times for solving the LP (ignore the ‘‘Heur.’’ line for the moment).

A Heuristic Making Life Easier for Randomized Rounding. As can be seen from the results presented so far, the different randomized rounding approaches usually find solutions that are not far from the optimum. We now propose and analyze a heuristic way to improve the performance.

The rough idea is simple. Having solved the linear relaxation of the ILP, we know the optimal (relaxed) congestion C^* that can be achieved. The congestion we end up with stems from this C^* plus possible rounding errors inflicted in the congestion constraints (1). It is clear that randomized rounding has a higher chance to increase the congestion if there are many congestion constraints satisfied with equality in the relaxation.

Therefore, the heuristic we suggest is to resolve the LP with the following modifications. Let $\delta \in [0, C^*]$ be a parameter open for fine-tuning. We replace the congestion constraints (1) by $\forall e \in E : \sum_{i=1}^k x_{ie} \leq C^* - \delta + z_e$, where C^* is the (fixed) optimal congestion obtained from the first LP and $z_e \in [0, \delta]$ are new variables. The new objec-

	$5 \times 5, 10$	$10 \times 10, 10$	$15 \times 15, 10$	$5 \times 5, 75$	$10 \times 10, 75$	$15 \times 15, 75$
Optimum	3.37	2.19	1.98	14.76	7.76	5.39
RR [20]	+9.23%	+32.17%	+42.29%	+5.96%	+24.48%	+45.45%
RR [4]	+7.13%	+30.43%	+40.31%	+4.13%	+19.07%	+38.78%
RR+	+6.35%	+31.85%	+40.26%	+2.64%	+12.50%	+31.91%
DeRR [4]	+3.77%	+22.33%	+23.42%	+1.90%	+10.95%	+25.97%
DeRR [5]	+2.76%	+16.38%	+13.76%	+0.88%	+8.38%	+17.07%
DeRR+	+1.19%	+11.81%	+16.59%	+0.47%	+4.25%	+14.84%

Table 1: Congestions achieved by the 7 different approaches for a selection of 6 instances. The optimum was computed by solving the IP via CPLEX (not feasible for larger instances). For the other algorithms we state the relative increase of the congestion over the optimum.

tive is to minimize $\sum_{e \in E} z_e$. Since the z_e are at most δ , the flow given by a solution of this new LP also yields a congestion of at most C^* . However, the new objective punishes edges with total flow exceeding $C^* - \delta$. In consequence, the solution we obtain is also a solution for the original LP, but one that in addition tries to keep some room in the congestion constraints.

For the few instances that space permits do give data, the experimental results are again presented in Table 1. Line 4 contains the results obtained by using randomized rounding as in [4] after applying the heuristic and line 7 does so with the derandomization of [20,5]. We did the same experiments with the other two rounding algorithms. Since the results were mainly inferior (to a similar extent as without the improvement), we omitted these numbers in the table. In all experiments, we chose $\delta = 1$.

The results clearly show that using this heuristic can be worth the extra effort of solving a second LP. Apart from two instances with very small objective values 1.98 and 2.19, the heuristic always gains us a significant improvement. Surprisingly, these gains tend to be higher when using the derandomized rounding algorithm.

It should be noted, though, that solving the second LP can be costly, as the numbers in the last line of Table 2 indicate.

	$5 \times 5, 10$	$10 \times 10, 10$	$15 \times 15, 10$	$5 \times 5, 75$	$10 \times 10, 75$	$15 \times 15, 75$
IP (CPLEX)	0.0270	0.368	5.72	0.776	61.52	7977
LP (CPLEX)	0.0227	0.235	1.61	0.697	34.02	2004
Heur.	0.0129	0.612	9.42	0.096	51.31	1755
DeRR [5]	0.0009	0.0061	0.0178	0.0028	0.018	0.07
DeRR [4]	0.0126	0.1062	0.3332	0.0589	0.459	1.61

Table 2: Run-times of the 5 algorithms in seconds. Given is the time for this particular step. For example, the run-time of what is called “DeRR+” in Table 1 is the sum of the values in lines “LP”, “DeRR” and “Heur.”.

3 Max Coverage: Greedy, Rounding, and Hybrid Approaches

Another problem where dependent rounding has found application is the max-coverage problem. In this problem, the input is a set $\{S_1, \dots, S_n\}$ of sets and a budget bound L . The task is to select a set of L sets to maximize the size of their union. Additionally, there can be costs c_i associated with the sets, and weights or profits w_i associated with the elements. In this case the task is to maximize the weighted sum of the covered elements, subject to the constraint that the total cost of the sets is at most L . Approximation algorithms for max-coverage have been devised using both greedy and rounding-based approaches; see [3,10,20,1] for details.

The rounding-based approximations start from the following LP-relaxation of the problem. Let n be the number of sets, and m the number of elements in the instance. Use variables $x \in [0, 1]^m$ and $y \in [0, 1]^n$; we maximize $\sum_{i=1}^m w_i x_i$ subject to (i) $\sum_{i=1}^n c_i y_i \leq L$, and (ii) $x_i \leq \sum_{j:i \in S_j} y_j$ for every $i \in [m]$. Let (x^*, y^*) be an optimal solution to this, of value W^* , and consider the expected outcome of independently rounding the variables y^* :

$$F(y^*) = \sum_{i=1}^m w_i (1 - \prod_{j:i \in S_j} (1 - y_j^*)). \quad (5)$$

In the unit-cost case, applying randomized rounding to y^* with a cardinality constraint preserving the sum $\sum_i y_i^*$ produces a rounding with expected value $F(y^*)$, due to the negative correlation properties of the rounding [20]; since $F(y^*) \geq (1 - 1/e)W^*$, we get a randomized $(1 - 1/e)$ -approximation. The derandomized version works via the method of conditional expectation. As shown by Ageev and Sviridenko [1], we can use $F(y)$ directly as a guide for the derandomization, and produce a rounding $y \in \{0, 1\}^n$ of y^* , such that $F(y) \geq F(y^*)$ with certainty.

For the case of weighted (knapsack) budget constraints, Srinivasan gives a rounding procedure (Lemma 3.1 in [20]) that approximately preserves the value of a weighted sum of the rounded variables, while guaranteeing negative correlation properties as in the unit-cost case. Combined with an enumeration and guessing phase, this provides a $(1 - 1/e + \varepsilon)$ -approximation for any $\varepsilon > 0$ [20]. Unfortunately, due to the inexactness of the budget bound, this phase becomes very expensive; we complement this by a budget-preserving rounding procedure, described below.

Algorithms and Improvements. The algorithms we mainly compare will be the greedy algorithm, and variations on the rounding-based algorithms. The *greedy algorithm* repeatedly selects a set fitting in the budget that maximizes the ratio of the profit of the newly covered elements to the cost of the set. The *rounding-based* approach is outlined above. Recall that the expected value of a single randomized rounding equals $F(y^*)$, and can thus (unlike in Section 2) be computed exactly. We consider three ways of boosting this value. The first, *random-1000*, is to simply apply randomized rounding 1000 times and pick the best result; the second is *derandomization*, using Srinivasan-type rounding directly on $F(y)$, with arbitrary order of variable comparison. The third is *gradient-based rounding* which works as follows.

Recall that cardinality-preserving randomized rounding works by repeatedly considering pairs of non-integral variables and readjusting their values, maintaining the

sum, such that one of them becomes integral (see e.g. [5]). By gradient-based rounding, we attempt to identify the best pair of variables to select for adjustment in each step. To truly find this pair would require $O(n^2)$ comparisons, each with cost $O(m)$, but we can approximate the selection by considering the gradient of $F(y)$. It is easy to show that if y_i and y_j are non-integral, and if $\frac{\partial F(y)}{\partial y_i} \geq \frac{\partial F(y)}{\partial y_j}$, then moving mass from y_j to y_i will keep the value of $F(y)$ non-decreasing. Thus we only need to compute and update the partial derivatives $\frac{\partial F(y)}{\partial y_i}$, which can be done analytically at a cost of $O(nm)$ per step, and we can in every step pair off the variables with largest and smallest values of partial derivative. While the total complexity of the rounding process becomes $O(n^2m)$, as opposed to $O(nm)$ for standard derandomized rounding, the time requirement is small in practice (see experiments below).

Finally, we introduce the following *budget-preserving rounding*.

Theorem 1. *Let $y \in [0, 1]^n$ such that $\sum_i c_i y_i = L$. In polynomial time, one can compute a rounding $\tilde{y} \in \{0, 1\}^n$ such that $\sum_i c_i \tilde{y}_i \leq L + \max_i c_i$ and $F(\tilde{y}) \geq F(y)$.*

Proof sketch. Instead of maintaining the cardinality in each rounding step, change the variables to be rounded by unequal amounts, maintaining the weighted budget. It is easy to show that the convexity property used in [1] to derandomize the unit-cost case still holds for $F(y)$ over the new operation. \square

For problems with a weighted budget, we use the same three rounding methods as for the unweighted case, with the random-1000 and the derandomized roundings using Lemma 3.1 of [20], and the gradient-based rounding using Theorem 1. Should a rounding exceed the budget constraint, we will greedily discard sets until the budget bound is reached.

A Unit Disk Maximum Domination PTAS. As a source of real-world instances, we consider a type of max-coverage instances derived from planar point set data. Given a set of points $P = \{p_1, \dots, p_n\}$ in the plane and a *diameter* d , we define a graph (the unit disk intersection graph) by letting two points p_i, p_j be connected if and only if the Euclidean distance between them is at most d . In this graph, we consider the problem of max-domination, where selecting a vertex v covers v and all its neighbours. Interpreted as max-coverage, we thus get an instance where every vertex corresponds to one set and one element. All sets will have unit cost; the elements may have weights, interpreted as the profit of covering them.

This problem is NP-hard, as follows from the hardness of Minimum Dominating Set in unit disk graphs [15]. However, it has good approximation properties—using the grid-based shifting strategy of Arora [2], we are able to provide a polynomial-time approximation scheme (PTAS). This strategy was also applied to a related problem on the placement of wireless base stations by Glaßer et al. [8]. The proof is omitted for lack of space.

Theorem 2. *For any $\ell > 1$, the Max Domination problem on unit disk graphs with weighted vertices and unit costs admits a $(1 - 2/\ell)$ -approximation in time $n^{O(\ell^2)}$.*

In our implementation, we replace certain steps by an MIP solver, specifically the exhaustive enumeration phase for the subproblems, and the dynamic programming

knapsack step. With these modifications, execution of the algorithm for non-trivial approximation ratios becomes possible. Unfortunately, we will see that even with these modifications, the PTAS approach is inferior to the greedy and rounding algorithms for realistic approximation settings.

Experimental Setup. Our instances are of two types. For the unit disk max-domination problem, we use benchmark instances stemming from a real-world facility location problem, previously used in [17] and available at [14]. For these instances, a demand is provided with every point; we use these demands as profit values. To complement this, we use instances converted from facility location problems; all such instances are downloaded from UfLib [21]. In both cases, we convert the instances to max-coverage by selecting an appropriate distance threshold for membership. Though some instances are weighted, in all tests the individual set costs will be much smaller than our allocated budget, avoiding potential issues with approximations.

We use CPLEX for LP- and ILP-solving; all other algorithms were implemented by the authors in C.

Experiments. We now report the results of our experiments. To begin with, we show the running times of the different methods on various instances in Table 3. Note that the LP solver once again uses a significant fraction of our running time, and that performing many random roundings becomes more costly than derandomization, due to the need to evaluate the objective value for each solution. The low numbers for the gradient-based rounding, as compared to the derandomization, can partly be explained by the gradient-based rounding being problem-specific, while the derandomization uses general-purpose code.

Table 4 shows results for some individual instances (described below). The first we want to highlight are the Chessboard and Finite Projective Plane (FPP) instances [12], which will serve to reveal the differences between the different approximation approaches. The Chessboard instances are instances on a chessboard with side $3k$, and essentially correspond to a planar packing problem; the FPP instances are similarly a kind of packing problem, but on a graph with more complicated structure. Since all instances of these classes have equivalent combinatorial structure, we use only instance per class in our experiments. In both cases, the budget is set at the most difficult setting, which turns out to be just where the LP-relaxation can cover all or almost all elements.

The results immediately show the reason to pursue hybrid greedy/rounding algorithms. For the Chessboard instance, the LP-optimum is already integral, and thus every LP-rounding-based algorithm discovers an optimal solution. On the other hand, these instances are difficult for the greedy algorithm, as a few early mistakes, when all sets

Instance	n	LP	Random-1000	Derand.	Grad. rounding	Greedy	IP
br818-400-30	818	0.36s	0.80s	0.11s	0.09s	0.33s	25s
kmed1-1k-37	1000	0.97s	1.46s	0.22s	0.25s	0.90s	> 3600s
MR1-060-16.5	500	0.36s	0.74s	0.05s	0.04s	0.14s	> 3600s

Table 3: Running times for various instances and algorithms; the times for the rounding methods exclude the time for solving the relaxation.

Name	Size	Budget	Greedy	LP: once	1000 derand	gradient	Optimum
Chessboard	144	16	130	144	144	144	144
FPP ($k = 17$)	307	17	290	200	210	230	290
br818-400	818	30	28054	22157	26199	27397	28448
kmed1-1k	1000	37	948	709	817	923	962
MR1-060	500	16500	1444	1179	1254	1402	1445

Table 4: Experiments on single instances. The data is averaged over 100 runs where the input data is randomly permuted. The column “LP: once” shows the expected outcome of a single randomized rounding; the following three columns show our three rounding methods. The optimum gives the best upper and lower bound achieved by an IP solver after one hour of running time.

seem equivalent, will hurt the end tiling. In the FPP instances, however, we see the opposite effect. Here, upon inspection, we find that the LP-optimum is a useless mix of taking an equal, small amount of almost every variable, leaving all the work of finding a good integral solution up to the rounding. On the other hand, the greedy algorithm performs very well here; runs with an ILP-solver show that it is at most one step away from the optimum. We find that our proposed hybrid, the gradient-based rounding, produces consistent top-quality results in both test cases.

We now focus on the unit disk max-domination problem, with instances as described previously. We select the largest instance, with 818 points inscribed in a box of sides 6395 by 3975, and use a distance threshold of 400. This was chosen as a good balance, as too small or too large values (e.g., 100 resp. 800) creates too simple instances. Figure 1 shows the behaviour of the main algorithms (excluding the PTAS) for this instance, as depending on the budget. Observe that the LP-rounding approach is very powerful for small budgets (up to 20), while further guidance is needed for larger budgets. The gradient-based LP-rounding, providing just such guidance, produces top values throughout, frequently better than either the greedy or the standard rounding algorithms. This instance also appears in Tables 4 and 3 under the name br818-400 or br818-400- L , where L is the budget bound. Another instance class of the same type is the k -median instances [21]. Here we use the one named 1000-10, with a threshold of 1000, occurring in Tables 4 and 3 as kmed1-1k or kmed1-1k- L . For concerns of clutter, the PTAS is not included in the figure, but its data is given separately in Table 5.

Instance	PTAS ($\ell = 3$)		PTAS ($\ell = 5$)		PTAS ($\ell = 7$)		Greedy	IP	
	Value	Time	Value	Time	Value	Time		Value	Time
br818-400-20	20742	1.3s	22857	9s	24129	69s	25247	26192	0.5s
br818-400-25	23008	1.2s	24683	9s	25308	71s	26907	27670	9s
br818-400-30	23951	1.3s	24909	10s	27270	74s	28054	28709	25s

Table 5: PTAS performance compared to the greedy algorithm and IP solver.

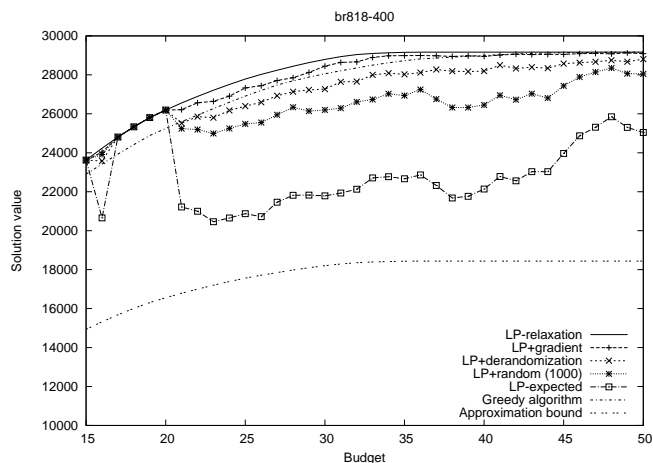


Figure 1: Results for unit disk max-coverage instance br818-400. The plot shows the value of the LP-relaxation, the outcome of the three rounding methods, and the expected value of a single rounding against the greedy algorithm. The approximation bound shows $(1 - 1/e)$ times the LP optimum.

Note that for every feasible setting, the PTAS is both of lower quality and significantly slower than the alternatives. The k-median-instance is omitted, since we lack point data.

Finally, we also consider an instance class with weighted budgets, namely the M* instances proposed in [13]. These instances have facility costs scaled so that facilities close to many customers are more expensive, a situation the authors of [13] propose would arise in real-world situations. In the data from [21], the distances were pre-scaled by demand values, making the distances inappropriate for our use; we remove this scaling, and instead use the demands as profit values. Table 4 gives the results for instance MR1 with distance threshold 0.6, under the name MR1-060. In general for this class, we found that the greedy algorithm and the gradient-based rounding method produce practically identical results, while the other methods are inferior to this.

A Greedy/LP Hybrid. Motivated by our results, we consider a different, more general form of greedy/LP hybrid. Before we commence with the LP-rounding, we allocate some portion of the budget to greedy pre-selection, and apply the LP-relaxation and rounding using the remaining budget to the thus reduced problem. We find best results with a pre-selection of between ten and forty percent, finding that this hybrid can produce results superior to either the greedy or the derandomized rounding algorithm on their own. Combining greedy pre-selection with gradient-rounding can produce further improvement on some instances (e.g., kmed1-1k), but for others (e.g., br818-400-30), no clear benefit was found. This data has been omitted for lack of space. We further report that with a pre-selection fraction of 0.3, both the Chessboard and the FPP instances of Table 4 receive optimal solutions.

4 Conclusions

We tried different randomized rounding approaches for routing and covering problems. We find that randomized rounding, in particular in derandomized versions, works well also for such non-artificial instances, yielding much better results in practice than what the theoretical guarantees assure. In addition, relatively simple fine-tunings give additional gains. This indicates a fruitful direction for further research.

References

1. A. A. Ageev and M. Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *J. Comb. Optim.*, 8:307–328, 2004.
2. S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *J. ACM*, 45:753–782, 1998.
3. G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms. *Management Science*, 23:789–810, 1977.
4. B. Doerr. Generating randomized roundings with cardinality constraints and derandomizations. In *STACS 2006*, pp. 571–583, 2006.
5. B. Doerr and M. Wahlström. Randomized rounding in the presence of a cardinality constraint. In *ALLENEX 2009*, pp. 162–174, 2009.
6. R. Gandhi, S. Khuller, S. Parthasarathy, and A. Srinivasan. Dependent rounding in bipartite graphs. In *FOCS 2002*, pp. 323–332, 2002.
7. R. Gandhi, S. Khuller, S. Parthasarathy, and A. Srinivasan. Dependent rounding and its applications to approximation algorithms. *J. ACM*, 53:324–360, 2006.
8. C. Glaßer, S. Reith, and H. Vollmer. The complexity of base station positioning in cellular networks. *Discrete Applied Mathematics*, 148:1–12, 2005.
9. R. M. Karp, F. T. Leighton, R. L. Rivest, C. D. Thompson, U. V. Vazirani, and V. V. Vazirani. Global wire routing in two-dimensional arrays. *Algorithmica*, 2:113–129, 1987.
10. S. Khuller, A. Moss, and J. S. Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70:39–45, 1999.
11. J. Kleinberg. *Approximation Algorithms for Disjoint Paths Problems*. PhD thesis, MIT, 1996.
12. Y. Kochetov and D. Ivanenko. Computationally difficult instances for the uncapacitated facility location problem. In *Proc. of the 5th Metaheuristic Conf. (MIC 2003)*, 2003.
13. J. Kratica, D. Toic, V. Filipovi, and I. Ljubi. Solving the simple plant location problem by genetic algorithm. *RAIRO Oper. Res.*, 35:127–142, 2001.
14. L.A.N.Lorena-instancias. <http://www.lac.inpe.br/~lorena/instancias.html>
15. S. Masuyama, T. Ibaraki, and T. Hasegawa. Computational complexity of the m -center problems on the plane. *Trans. of the Inst. of Electron. and Commun. Eng. of Japan. Sect. E*, E64:57–64, 1981.
16. OR-Library. <http://people.brunel.ac.uk/~mastjjb/jeb/info.html>.
17. M. A. Pereira, L. A. N. Lorena, and E. L. F. Senne. A column generation approach for the maximal covering location problem. *Int. Trans. in Oper. Res.*, 14:349–364, 2007.
18. P. Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *J. Comput. Syst. Sci.*, 37:130–143, 1988.
19. P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, 1987.
20. A. Srinivasan. Distributions on level-sets with applications to approximations algorithms. In *FOCS 2001*, pp. 588–597, 2001.
21. UflLib. <http://www.mpi-inf.mpg.de/departments/d1/projects/benchmarks/UflLib/>.